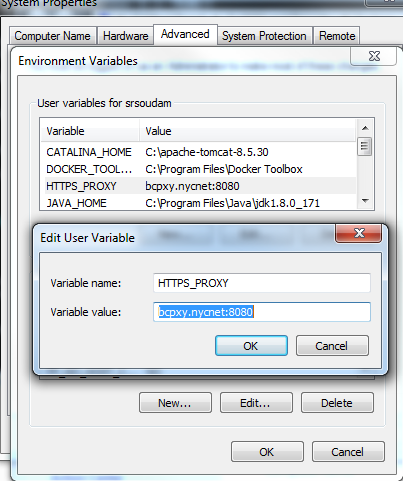
**How to Deploy Spring Boot Application to Cloud Foundry Platform**

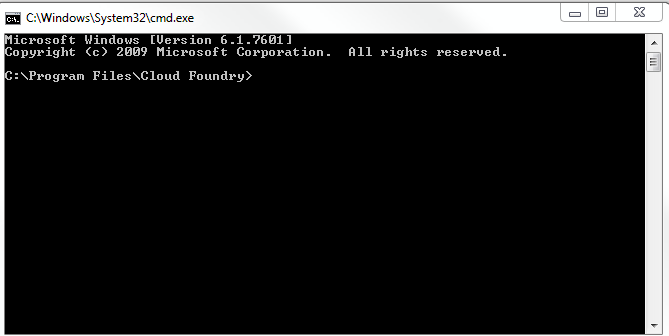
The cloud foundry works very well from command prompt and cloud foundry has provided one command line tool called cf which does almost all the activities for us. So to make this tool (cf command) available in local workstation, first we need to install and configure the *Cloud Foundry Command line (CLI) interface*.

1. Download the <https://docs.cloudfoundry.org/cf-cli/install-go-cli.html#windows> . It will prompt for the download. Save the zip file distribution.
2. Once installed go to folder installed C:\Program Files\Cloud Foundry in cmd and test it typing cf.

Troubleshooting:

If there is any https proxy issue when running the cf command, then set the **HTTPS\_PROXY** in the Environment Variables.





1. Once registered Sign into <https://login.run.pivotal.io/login>.

#### Login and logout from PWS Console using CLI

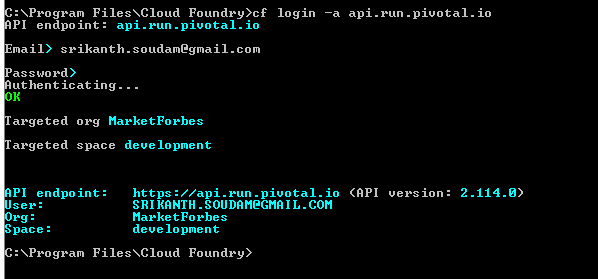
//To login

>> cf login -a api.run.pivotal.io

After this Enter username and password of PCF.

//To logout

 >> cf logout





## **Create Spring Boot Application**

1. Create Spring Boot Application using Maven project and add Dependencies of Web, Rest Repositories and Actuator.

@SpringBootApplication

**public** **class** SpringHelloWorldCFApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringHelloWorldCFApplication.**class**, args);

}

}

Create Controller Class as:

@RestController

**public** **class** SpringHelloController {

@RequestMapping("/display")

**public** String displayMessage(@RequestParam("name") String name) {

String message ="Hi "+ name + " .How are you today !";

**return** message;

}

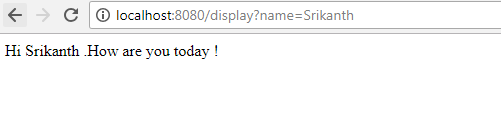
}

1. Add Context path and required properties in bootstrap.properties file in src\main\resources directory and add two properties there.

|  |
| --- |
| server.contextPath = /hello  management.security.enabled = false |
|  |

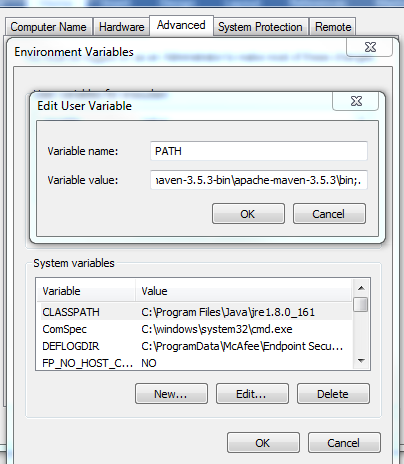
This will set one context path /hello for the application and management.security.enabled=false will disable security for management endpoints of spring boot like /env, /refresh etc.

1. Now run this as Spring Boot Application and test it locally on browser.

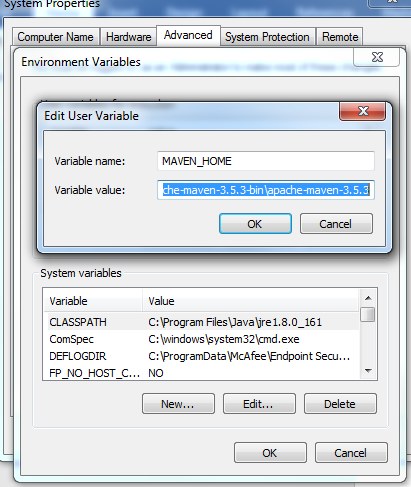


## **Deploy Spring Boot Application in Cloud Foundry Platform**

1. Generate the SpringCloudSample-0.0.1-SNAPSHOT(jar) file using Maven.
2. Install Maven and set the PATH as follows:



And set the MAVEN\_HOME as



**Troubleshooting:**

If you get any proxy issue after installing MVN and when you run mvn clean install (usually error like connect to repo.maven.apache.org 443 failed)

Add proxy settings in settings.xml in conf folder of MVN in the Proxies tag

<proxies>

<proxy>

<active>true</active>

<protocol>http</protocol>

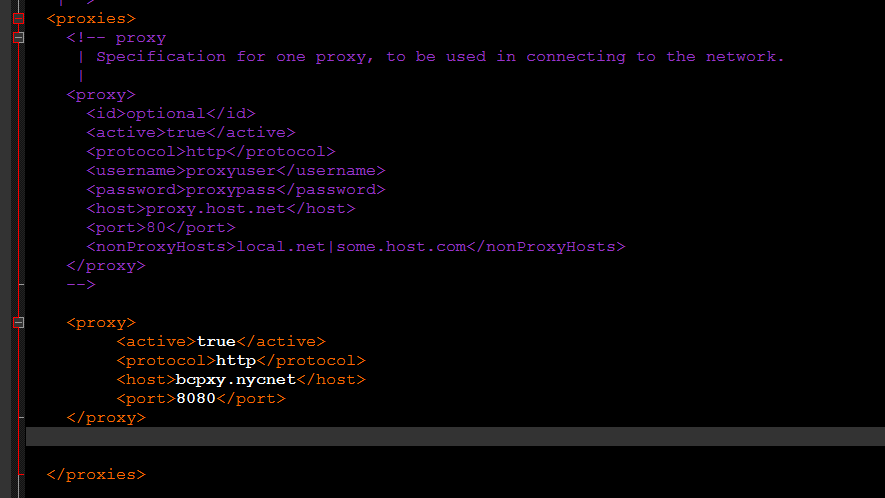
<host>bcpxy.nycnet</host>

<port>8080</port>

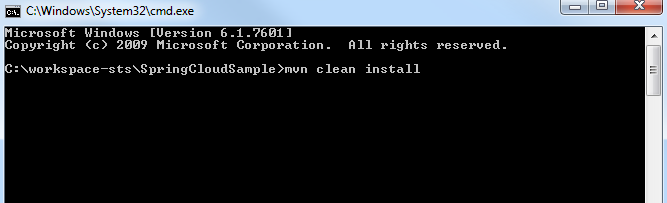
</proxy>

</proxies>

C:\apache-maven-3.5.3-bin\apache-maven-3.5.3\conf

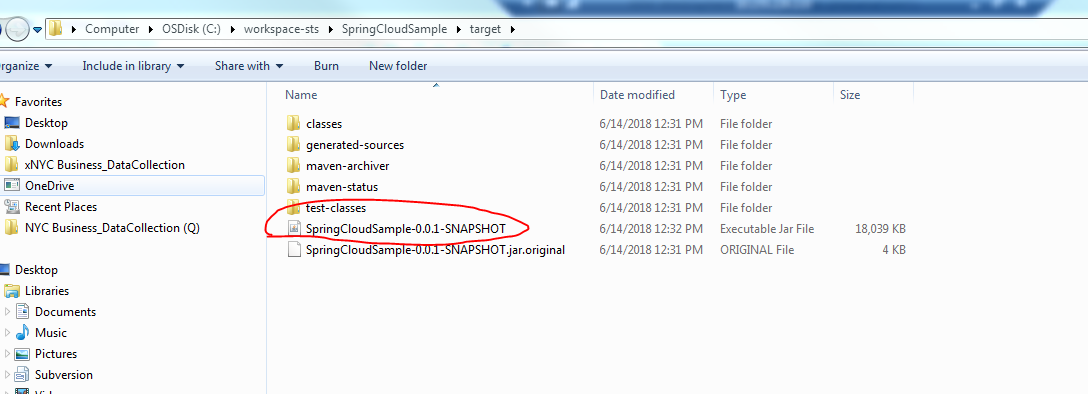


Now do mvn clean install using the following command by going to Project folder location:

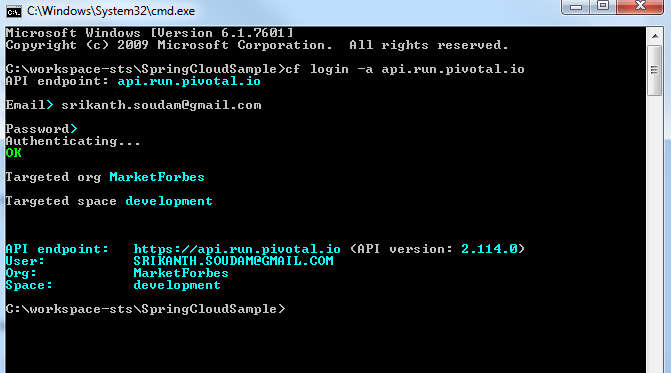


After that do mvn package.

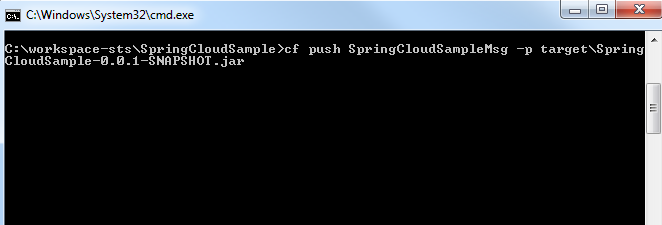
The generated jar will be placed in target folder of the application



Now login to PCF using the CLI as follows:

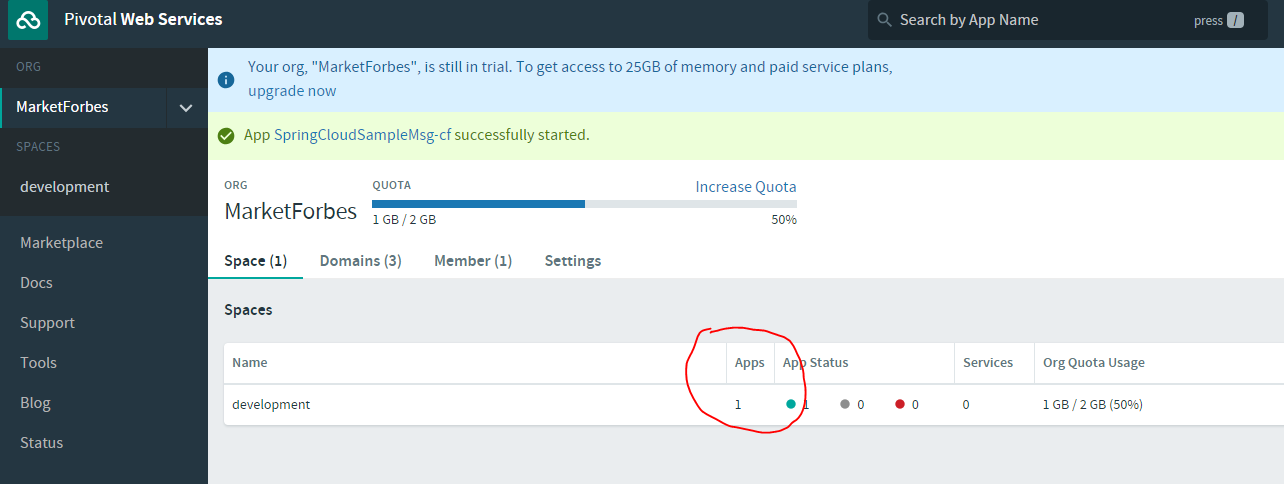


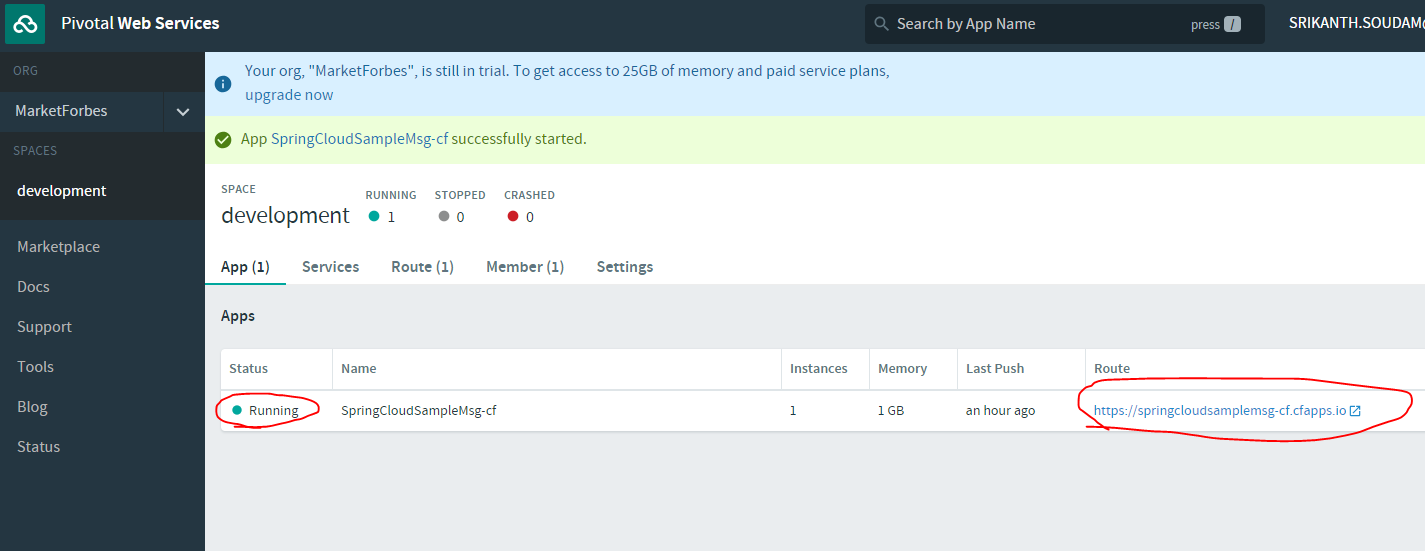
Now push the application jar into PCF as :



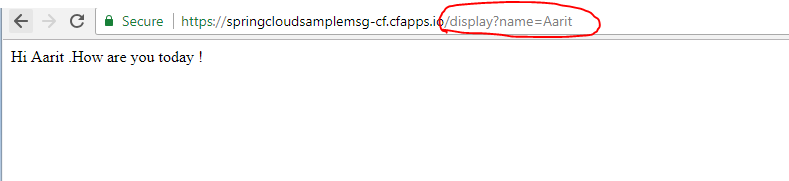
This will deploy the application into PWS(Pivotal Web Services) console.

You can Verify logging into PWS console.





This shows application is deployed and running on PWS.Now to test this click on the Route and add the following to URI.



**Reference Links:**

<https://howtodoinjava.com/spring/spring-cloud/pivotal-cloud-foundry-spring-boot-example/>

<https://docs.spring.io/spring-boot/docs/current/reference/html/getting-started-first-application.html>

<https://spring.io/blog/2011/01/17/green-beans-getting-started-with-maven-and-spring/>